Information Retrieval using a Database Management System in a Parallel Environment

LTC Kenneth L. Alford
U.S. Army
Fairfax, Virginia

Jim X. Chen
George Mason University
Fairfax, Virginia

Abstract

The ability to create documents is growing much faster than the ability to find and retrieve them. Information retrieval is the branch of information technology that deals with searching and retrieving relevant information. Information retrieval strategies, utilities, and evaluation methods applicable to this research are defined. Using commercial software and multiprocessor hardware available at the U.S. Army Research Laboratory's Major Shared Resource Center (ARL MSRC), Aberdeen, MD, an unclassified parallel relational database information retrieval system has been developed to process Web-based document collections. This research information retrieval system can provide several advantages over equivalent proprietary systems. Scalability, performance, filtering insights, and future research directions are also discussed.

Introduction

Anyone who has used a search engine on the Internet or an automated information retrieval system in a library knows that even the best retrieval systems available today often create frustration and waste time as irrelevant documents are retrieved for evaluation. The Department of Defense, together with other government organizations, academia, and business, continues to look for efficient ways to store and retrieve information.

Information retrieval (IR) is the branch of information technology that deals with searching and, more importantly, retrieving information relevant to user queries. Research in the area of automated information retrieval began nearly 40 years ago and has grown in recent years.

Problem Statement

In its simplest form, IR is concerned with finding the most relevant documents possible in a time period acceptable to the user. IR systems help users find the most relevant documents that match a given query. The information retrieval problem has been more precisely defined as:

> Given a document collection $D_1, D_2, \ldots D_n$, and a query Q, determine a similarity coefficient, $SC(Q,D_i)$, where $1 \le i \le n$, such that higher $SC(Q, D_i)$ values indicate greater relevance to the query [4].

This research seeks to demonstrate that a parallel relational database IR system: (1) produces precision performance results equivalent to other information retrieval systems, (2) is scalable over large document collections, and (3) benefits from the application of document filtering.

Resources

The target computer for this research is located at the U.S. Army Research Laboratory's Major Shared Resource Center (ARL MSRC), Aberdeen, MD.

Hardware

This research is being conducted on an unclassified Sun Enterprise 10000 computer—a scalable symmetric multiprocessing architecture. Each processor shares common access to common memory with the following system attributes:

- Twenty-eight 333 MHz UltraSPARC CPUs (configured into three domains — the domain used to support this research has 16 dedicated processors)
- 28.5 gigabytes (GB) of memory
- 500 GB of disk storage

Software

This research requires only the following commercial software for implementation: (1) Oracle 8 Enterprise Edition (parallel relational database management software), (2) standard C language compiler (with a math library), and (3) standard UNIX operating system utilities. This IR system could also be implemented using another parallel relational database management software program.

Data

In early 1997, the Internet Archive conducted a trawl of the World Wide Web [6]. The documents collected were lightly filtered—to remove documents larger than 2 megabytes (MB)—and the first 100.426 GB of document data were bundled as the Very Large Collection, Second Edition (VLC2). The VLC2 contains over 18.5 million Web documents. Each VLC2 document received a minimal set of SGML (standard generalized markup language) tags, marking the beginning and end of each document, the document number, and header information captured during the trawl. Figure 1 shows a sample document.

The VLC2 provides a frozen snapshot of the World Wide Web. There are very few formatting guidelines for documents in the VLC2 collection. The majority of the documents consist of English HTML (hypertext markup language) code and text, but the collection also contains numerous binary files (that were mislabeled on the Web as HTML) and foreign language documents in a wide variety of languages.

The quality and consistency of HTML code and text varies widely within the VLC2 collection. Improperly formatted HTML tags with missing brackets, typographical errors, incorrect tags, missing closing tags, etc., are common throughout the collection. The only unbroken rule appears to be that 'on the Web, there are no rules.'

The VLC2 contains two document collection subsets. The BASE1 subset is a 1% sample created by copying every 100th document bundle from the main VLC2 collection. The BASE10 subset collection is a 10% sample that copies every 10th document bundle from the main collection. Table 1 contains a summary of the VLC2, BASE10, and BASE1 document collections.

User queries, sometimes referred to as topics, may be created in many ways. The Text REtrieval Conference (TREC) standard for topics uses an SGML-tagged query format. Queries are numbered and contain a title, description, and narrative. Figure 2 shows a sample TREC topic regarding civilian deaths in Africa.

NIST provides 50 tagged topics for each annual conference. There are currently 400 topics. Topics from previous conferences are often used as training data to prepare for topics from the current research year.

Approach

When creating an information retrieval system, developers must answer two key questions. First, which retrieval strategy will be used to measure the similarity between documents and queries? And, second, which retrieval utility, or utilities, can be used to improve the efficiency of the set of retrieved documents?

IR Strategies

A retrieval strategy is the primary approach that an IR system takes to retrieve documents. Retrieval strategies are usually developed around the concept that the more indexed items that queries and documents share in common, then the more relevant an individual document may be to a specific query. Most retrieval strategies calculate a similarity measure between queries and documents. Numerous strategies exist to develop good similarity measures.

The vector space model, first proposed by Salton, Wong, and Yang [8], is one strategy for computing a similarity measure between a query and a document. A vector is used to represent the terms within a document. Queries are treated as small documents and are represented in the system by similarly constructed vectors. With queries and documents represented by term vectors, it is possible to calculate the similarity between both vectors.

The measure of relevance between query and document vectors is referred to as the similarity coefficient (SC). Traditionally, the similarity between vectors is measured by the size of the angle between them; less difference means greater similarity. However, any monotonic function representing

that angle will also suffice [4]. Conceptually, all document and query vectors contain a component for each unique term that occurs anywhere in the document collection. Document and query terms are usually stored case insensitive to simplify comparisons.

Binary-value vectors are of limited use in determining similarity because they are unable to represent how often terms appear within a document. Vector components can be easily extended to record term frequency (*tf*) values. In addition to including term frequency values, research has demonstrated [7] that terms appearing frequently in a document collection are poor discriminators of query relevance, so they should receive a lower weight than terms that appear infrequently.

Term weights based on the frequency that a term appears within the entire document collection can also be added to document vector components. This concept, called the inverse document frequency (*idf*), is an important element of many IR systems. Vector component term weights may also be calculated using a combination of term frequency and inverse document frequency measures—a *tf-idf* measure.

The following definitions are useful to create a document vector [4]:

n	=	**Number of Unique Terms** (in the document collection)
d	=	**Total Documents** (in the document collection)
tf_{ij}	=	**Term Frequency** (number of occurrences of term t_j in document D_i)
df_j	=	**Document Frequency** (the number of documents that contain term$_j$, t_j)
idf_j	=	**Inverse Document Frequency** (log (d / df_j))

The weighting factor (w) for each term, j, in a document, i, is the product of the term frequency (*tf*) and the inverse document frequency (*idf*):

$$w_{ij} = tf_{ij} \diamondsuit idf_j$$

Similar weights are calculated for query terms, and an SC is calculated between the query and each document in the collection.

In addition to the vector space model, other information retrieval strategies have been developed. They include: (1) probabilistic retrieval, (2) inference networks, (3) extended Boolean, (4) latent semantic indexing, (5) neural networks, (6) fuzzy set retrieval, and (7) genetic algorithms. (See Grossman and Frieder [4] for a more complete list.)

IR Utilities

Retrieval utilities fine-tune the precision of an IR strategy and are generally independent of the retrieval strategy chosen. Retrieval utilities must be carefully integrated into the IR strategy in order to complement the basic strategy used. Several IR utilities have proven helpful with this research, and we will briefly discuss two, parsing and stop lists. Several other IR utilities are also available.

Parsing is the process of identifying tokens contained within a document or text collection, and it is one of the first steps of almost every IR system. Numerous parsing methods exist. Phrase creation, as used in this research, is the process of combining two or more single word terms into multiword phrases that occur within the original document. A user entering the query “Pearl Harbor,” for example, would not wish to retrieve a document containing the statement “He purchased a beautiful pearl for her near the harbor.”

A stop list, or “negative dictionary” as it is referred to in Fox [1], contains terms that are not useful to store in an index. Stop-list terms ("stopwords") can be removed during the parsing of a document collection.

Several categories of stopwords may be added to a stop list; the two most common kinds of stopwords are: (1) terms that have little semantic content, and (2) terms that appear with such a high frequency in a document collection they cannot contribute to discriminating between documents. One study, for example, showed that the 10 most common words in English — *the, of, and, to, a, in, that, is, was,* and *he* — account for one-fourth of all written text [2]. These words appear on virtually every stop list. The creation of stop lists currently involves as much art as science. Even structured attempts at creating stop lists involve many arbitrary decisions regarding the inclusion or exclusion of specific terms.

Measuring Results

The accuracy of an information retrieval system is measured through the calculation of two document retrieval ratios: *precision* and *recall*. Precision asks, “How many of the documents found were relevant to the query?” Recall asks, “How many relevant documents were found?”

Relevance ranking is the concept that retrieved documents are listed in priority order with the most relevant documents listed first. The initial relevant documents retrieved by an IR system are usually easy to find; but to retrieve all relevant documents in a collection is far more challenging.

Before 1992, there were no widely used standards to judge the effectiveness of information retrieval algorithms, strategies, and utilities. At that time, the Information Technology Office of the Defense Advanced Research Project Agency (DARPA) and the National Institute of Standards and Technology (NIST)—with participation from industry, academia, government, and business—teamed together to encourage information retrieval research. The goals of TREC are to: (1) encourage research in text retrieval based on large text collections, (2) create an open forum for an exchange of ideas between government, academia, and industry, (3) speed technology transfer from research into commercial products, and (4) to create and make available appropriate evaluation techniques [9].

TREC provides standardized research data, queries, relevance judgments, and uniform scoring procedures. NIST sponsors an annual conference to encourage continued progress in IR. TREC has become a standard forum to evaluate and compare IR strategies, utilities, and systems. Currently in its eighth year, TREC is a year-long research effort that attracts participation from teams scattered throughout the world. A portion of this research will be submitted to the 1999 TREC.

Relational Database Information Retrieval

An IR system can be constructed using a relational database management system (DBMS) [3], and there are several advantages for doing so:

- The system requires no proprietary software or hardware that would cause life-cycle costs to be substantially less than equivalent proprietary systems.
- Relational database programs are readily available.
- The system requires no special training other than general database knowledge and familiarization with IR concepts.
- The system uses unchanged structured query language (SQL) and is portable to other relational database programs.

Implementation

Relational DBMS IR systems can be constructed several ways. This implementation generally follows the organization proposed by Grossman and Frieder [3] and Grossman [4] and contains the following database relations:

DOCUMENT_TERMS_TABLE (Document_ID, TermCount, Term) — to store terms from all documents in the collection (one row per term)

DOCUMENT_TABLE (Document_ID, Document_Name, Average_Term_Frequency, Distinct_Term_Count, Sum_Term_Frequency, Log_Average_Term_Frequency) — to store document-level information and statistics about the document collection (one row per document)

INVERTED_INDEX_TABLE (Term_Frequency, Document_Frequency, Normalized_Inverse_Document_Frequency, Term) — to store an inverted index

QUERY_TERM_TABLE (Query_ID, Term_Count, Normalized_Inverse_Document_Frequency, Term) — to store query information (one row per query)

RESULTS_TABLE (Query_ID, Document_ID, Document_Name, Similarity_Coefficient) — to store information regarding retrieved documents (one row per document)

Many IR systems, including the vector space model, implement an inverted index to improve efficiency. An inverted index contains one entry—or, in the case of a relational DBMS, one row—for each document term stored in the database.

Parallel IR systems have also been created using relational database systems [7]. If an SQL statement runs on multiprocessor hardware, many relational DBMS environments, such as Oracle, can perform parallel processing with no changes to the original SQL code.. It is possible to control the parallel performance of numerous standard database operations, such as load, insert, and update. Parallel relational database systems may also be fine-tuned by providing specific “parallel hints” to the database optimizer and compiler.

Parallel database IR systems have been successfully constructed on the Sun 10000 computer for the BASE1 (1 GB) and BASE10 (10 GB) document collections. Prior to loading the BASE1 and BASE10 document collections in Oracle, a parsing preprocessor (written in C) was rewritten to parse the unstructured Web-based documents in the VLC2 collection.

Insights Gained

This research is ongoing, but encouraging preliminary results have been obtained regarding performance, scalability, and filtering.

Performance

Precision results were calculated using the standard TREC evaluation software and relevant document judgment data set provided by TREC. The BASE10 parallel database IR system developed during this research achieved a P@20 (average precision of the first 20 documents) evaluation of 0.2728 when topic titles were evaluated. When topic titles and descriptions were both used, the P@20 value rose to 0.3388. Average precision values take into account the position ranks of relevant documents.

In an effort to compare TREC retrieval systems with commercial Web search engines, TREC-7 "title only" queries were given to five well-known search engines [5]. The same judge who determined relevancy for TREC submissions evaluated the search engine results. The search engines were not penalized for returning "dead" or duplicate links. The search engine relevancy results are shown at Table 2. The BASE10 parallel database IR system performed better or equivalent to three of the five search engines tested.

Scalability

In addition to efficiency, IR systems are measured by speed in retrieving documents. The initial scalability results have been mixed.

All 50 TREC-7 query topics can be executed in parallel (using 10 background processes each containing 5 separate queries) on the BASE1 document collection in 14 minutes and 40 seconds. Using the same SQL code, those same queries execute in parallel on the BASE10 (10 GB) document collection in 4 hrs. and 13 min. (an increase 73% larger than linear growth would predict, based on the increase in document size between the BASE1 and BASE10 collections).

Substantial reductions in storage space, though, have been achieved using partitioned tables and local partitioned bitmap indexes. By using a local bitmap index, for example, a nonpartitioned, standard 1 GB index in the BASE1 document collection was reduced to 2.775 GB in the BASE10 document collection (even though it was indexing 10 times more information).

Several parallel database and operating system features hold promise for decreasing system creation and query execution times:

- Document collections can be preprocessed in parallel by using mutually exclusive document collection subsets and running multiple copies of executable code as separate background processes.

- When parallel-aware SQL code is run on multiprocessor hardware, the database engine can parallelize the data set and the SQL command involved. Common database functions—such as, loading data tables, inserting, updating, and deleting rows—will be automatically run in parallel by the database systems if parallel options have been enabled. Equal data partitions should also improve efficiency.

- SQL scripts can be run in parallel from the operating system prompt as multiple background processes. The VLC2 document collection can be logically or physically partitioned into several document subsets. Each subset can be loaded into its own IR system. Queries can be executed in parallel over the entire document collection with results being merged at the conclusion to form the set of retrieved documents.

Filtering

The frequency that stopwords occur within a document may be used as a filter to identify and remove nontext and foreign language documents that randomly exist within a document collection.

Initial stop list document filtering heuristics—based on the number and kinds of words stopped—have been developed and are being fine-tuned. Stop list document filtering heuristics successfully identified and removed 22,489 documents from the BASE1 document collection (12 percent of the entire collection). None of the documents removed were relevant to any of the 50 query topics from last year's TREC research effort. Numerous checks of the documents removed failed to find any English documents that had been incorrectly extracted from the document collection.

Future Directions

Several extensions to this research are currently underway:

- Creating an IR system using the entire VLC2 100 GB Web document collection.

- Increasing the proportion of processing that is performed in parallel by fine-tuning parallel parameters, hints, and data structures.

- Improving the use of stop-list information as a document filter.

- Incorporating Web link information in a parallel database IR system in order to increase retrieval precision. A separate 2-GB subset of the VLC2 document collection has been identified for use during the 1999 research year. A connectivity server is also being provided to assist in the evaluation of Web-link information.

Progress is being made, but much is left to discover and improve.

Acknowledgments

LTC Alford wishes to thank Anthony Pressley and the staff and consultants, especially Alex Balboa and Jeff McIsaac, at the ARL MSRC, Aberdeen, MD for their generous support in making this research possible.

References

[1] C. Fox, "A Stop List for General Text", *SIGIR Forum*, Vol. 24:1-2, 1990, pp.19–35.

[2] W. Francis and H. Kucera, *Frequency Analysis of English Usage. Lexicon and Grammar*. Houghton Mifflin: New York, 1982.

[3] D. Grossman, *Integrating Structured Data and Text: A Relational Approach*, doctoral dissertation, George Mason Univ., School of Information Technology and Engineering, 1995.

[4] D. Grossman and O. Frieder, *Information Retrieval Algorithms and Heuristics*. Kluwer Academic Publishers: Boston, 1998.

[5] D. Hawkings, N. Craswell, P. Thistlewaite, and D. Harman, "Results and Challenges in Web Search Evaluation, *Eighth Int'l World Wide Web Conf. (Toronto)*, 1999.

[6] The Internet Archive, http://www.archive.org (current April 1999).

[7] C. Lundquist, D. Grossman, and O. Frieder, "Improving Relevance Feedback in the Vector Space Model," *Proc. of the Sixth ACM Annual Conf. on Information and Knowledge Management (CIKM '97)*, ACM Press, New York, 1997.

[8] G. Salton, A. Wong, and C. Yang, "A Vector Space Model for Automatic Indexing," *Comm. of the ACM*, Vol. 18, No. 11, 1975, pp. 613–620.

[9] E. Voorhees and D. Harman, "Overview of the Sixth Text REtrieval Conf. (TREC-6)", *Proc. of the Sixth Text REtrieval Conf. (TREC-7)*, U.S. Dept. of Commerce, 1998, p. 24.

Figure Captions

<DOC>
<DOCNO>lA000-000168-B000-00</DOCNO>
<DOCHDR>
http://hilbert.anu.edu.au:80/~david/z80.html 150.203.43.6 19970106002742
text/html 1804
HTTP/1.0 200 OK
Date: Monday, 06-Jan-97 00:27:40 GMT
Server: NCSA/1.3 MIME-version: 1.0
Content-type: text/html
Last-modified: Wednesday, 04-Sep-96 04:09:00 GMT
Content-length: 1621
</DOCHDR>
<head>
<title>John Steven's Z80 Page</title>
</head><body>
<H1>Introduction</H1>
A small group of dedicated hackers are working on a C cross-compiler
for the >>Z80<< (and a number of other small micros)
<P </body>
</DOC>

Figure 1. Sample VLC2 document.

<num> Number: 306
<title> African Civilian Deaths
<desc> Description:
How many civilian non-combatants have been killed
in the various civil wars in Africa?
<narr> Narrative:
A relevant document will contain specific casualty
information for a given area, country, or region.
It will cite numbers of civilian deaths caused
directly or indirectly by armed conflict.

Figure 2. Sample TREC SGML-formatted topic.

Tables

VERY LARGE COLLECTION, SECOND EDITION (VLC2) STATISTICS

MAIN (100%) Collection Statistics

```
Size of Entire Collection   100.426 GB
Number of Different Hosts   116,102
Hosts with Only One Page    24,814
Approximately 160 pages contributed per host (on average)
                                Size Range          Average Size
          97 Collections    0.382 GB - 1.064 GB       1.035 GB
      50,023 Bundles        0.002 MB - 4.000 MB       2.056 MB
  18,571,671 Documents      0.199 KB - 3823.749 KB    5.670 KB
```

BASE10 (10%) Collection Statistics

```
                                Size Range          Average Size
           1 Collection     Size = 10.055 GB
       5,003 Bundles        0.002 MB - 3.950 MB       2.058 MB
   1,857,405 Documents      0.236 KB - 2822.490 KB    5.677 KB
```

BASE1 (1%) Collection Statistics

```
                                Size Range          Average Size
           1 Collection     Size = 1.005 GB
         501 Bundles        0.441 MB - 3.315 MB       2.055 MB
     187,381 Documents      0.259 KB - 2067.306 KB    5.625 KB
```

Table 1

A Comparison of Precision Results for 5 Commercial Search Engines and a Parallel Database IR System

Search Engine	#1	#2	#3	#4	#5	Parallel Database IR System
P@20 (Title only)	**0.306**	**0.288**	**0.231**	**0.377**	**0.289**	**0.273**

Based on relevancy judgments made by the 1998 TREC-7 evaluation team.

Table 2